EMBEDDED
LINUX
CONFERENCE

ELC-E Dublin 2022

Slides: **tinyurl.com/elce22-riscv**

Baylibre

Linux on RISC-V

Drew Fustini <dfustini@baylibre.com>

$ whoami

- Linux kernel developer, BayLibre
  - embedded software consultancy based in Nice, France, with ~50 engineers around the world contributing to open source projects like Linux, U-Boot, Android and Zephyr
- Board of Directors, BeagleBoard.org Foundation
- Board of Directors, Open Source Hardware Association (OSHWA)
  - OSHW Certification Program
- Ambassador, RISC-V International

RISC-V: a Free and Open ISA

- Started by a computer architecture research group at University of California Berkeley in 2010 led by Krste Asanovic
- **V** as in the roman numeral five, because it is the 5th **RISC** instruction set to come out of UC Berkeley
- **Free and Open** because the specifications are published under an open source license: Creative Commons Attribution 4.0 International
  - Volume 1, Unprivileged Spec v. 20191213 [PDF]
  - Volume 2, Privileged Spec v. 20211203 [PDF]

What is different about RISC-V?

- Simple clean-slate design
  - Avoids any dependencies on microarchitecture style *(in-order, out-of-order, etc)*
- Modular design
  - Suitable for everything from microcontrollers to supercomputers
- Stable base
  - Base integer ISAs and standard extensions are frozen
  - Additions via optional extensions, not new versions

(source: Instruction Sets Want to be Free, Krste Asanovic)

RISC-V base integer ISAs

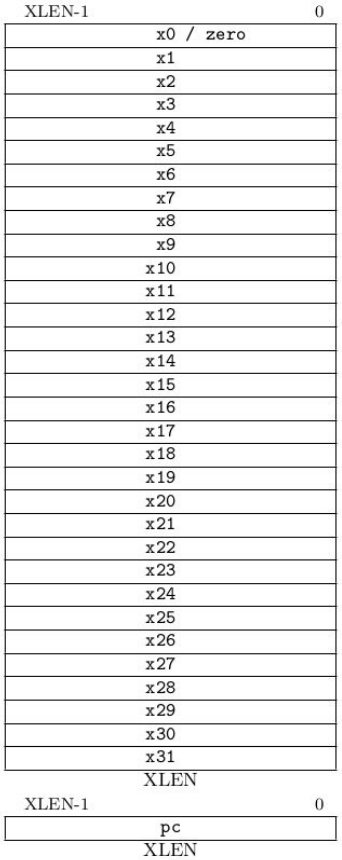
- RV32I: 32-bit
  - less than 50 instructions needed!
- RV64I: 64-bit
  - Most important for Linux
- RV128I: 128-bit
  - Future-proof address space

imm[31:12]			rd	0110111	LUI
imm[31:12]			rd	0010111	AUIPC
imm[20|10:1|11|19:12]			rd	1101111	JAL
imm[11:0]		000	rd	1100111	JALR
imm[12|10:5]	rs2	000	imm[4:1|11]	1100011	BEQ
imm[12|10:5]	rs2	001	imm[4:1|11]	1100011	BNE
imm[12|10:5]	rs2	100	imm[4:1|11]	1100011	BLT
imm[12|10:5]	rs2	101	imm[4:1|11]	1100011	BGE
imm[12|10:5]	rs2	110	imm[4:1|11]	1100011	BLTU
imm[12|10:5]	rs2	111	imm[4:1|11]	1100011	BGEU
imm[11:0]		000	rd	0000011	LB
imm[11:0]		001	rd	0000011	LH
imm[11:0]		010	rd	0000011	LW
imm[11:0]		100	rd	0000011	LBU
imm[11:0]		101	rd	0000011	LHU
imm[11:5]	rs2	000	imm[4:0]	0100011	SB
imm[11:5]	rs2	001	imm[4:0]	0100011	SH
imm[11:5]	rs2	010	imm[4:0]	0100011	SW
imm[11:0]		000	rd	0010011	ADDI
imm[11:0]		010	rd	0010011	SLTI
imm[11:0]		011	rd	0010011	SLTIU
imm[11:0]		100	rd	0010011	XORI
imm[11:0]		110	rd	0010011	ORI
imm[11:0]		111	rd	0010011	ANDI
0000000	shamt	001	rd	0010011	SLLI
0000000	shamt	101	rd	0010011	SRLI
0100000	shamt	101	rd	0010011	SRAI
0000000	rs2	000	rd	0110011	ADD
0100000	rs2	000	rd	0110011	SUB
0000000	rs2	001	rd	0110011	SLL
0000000	rs2	010	rd	0110011	SLT
0000000	rs2	011	rd	0110011	SLTU
0000000	rs2	100	rd	0110011	XOR
0000000	rs2	101	rd	0110011	SRL
0100000	rs2	101	rd	0110011	SRA
0000000	rs2	110	rd	0110011	OR
0000000	rs2	111	rd	0110011	AND
fm | pred | succ		000	rd	0001111	FENCE
000000000000		000	00000	1110011	ECALL
000000000001		000	00000	1110011	EBREAK

(source: RISC-V Summit 2019: State of the Union, Krste Asanovic)

RISC-V base integer registers

- XLEN defines the register width
  - XLEN=32 for RV32I
  - XLEN=64 for RV64I
- 32 registers named x0 to x31
- Dedicated PC register
- Base ISA talk by Andrew Waterman explains the instruction encoding scheme



(source: Figure 2.1: RISC-V base unprivileged integer register state)

RISC-V ABI

- x1 to x31 are all equally general-use registers as far as the processor is concerned
- RISC-V psABI defines standard functions for these registers *[PDF]*
  - s0 to s11 are preserved across function calls
  - argument registers a0 to a7 and the temporary registers t0 to t6 are not

Register	ABI Name	Description	Saver
x0	zero	Hard-wired zero	—
x1	ra	Return address	Caller
x2	sp	Stack pointer	Callee
x3	gp	Global pointer	—
x4	tp	Thread pointer	—
x5	t0	Temporary/alternate link register	Caller
x6–7	t1–2	Temporaries	Caller
x8	s0/fp	Saved register/frame pointer	Callee
x9	s1	Saved register	Callee
x10–11	a0–1	Function arguments/return values	Caller
x12–17	a2–7	Function arguments	Caller
x18–27	s2–11	Saved registers	Callee
x28–31	t3–6	Temporaries	Caller

(source: RISC-V Assembly Programmer's Manual)

RISC-V Standard Extensions

- **M**: integer multiply/divide
- **A**: atomic memory operations
- **F**, **D**, **Q**: floating point, double-precision, quad-precision
- **G**: "general purpose" ISA, equivalent to **IMAFD**
- **C**: compressed instructions conserve memory and cache
- Most Linux distributions target **RV64GC**

(source: RISC-V Summit 2019: State of the Union, Krste Asanovic)

Ratified in 2021

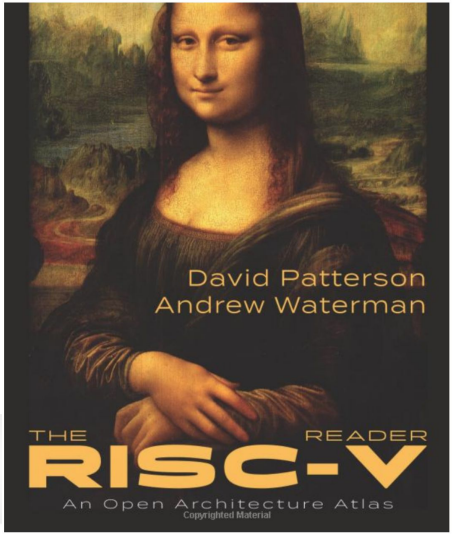
- 15 new specifications representing more than 40 extensions
- Vector
- Hypervisor
- Scalar Cryptography
- Bit Manipulation

RISC-V Profiles

- RISC-V is a highly modular and extensible architecture
  - Flexibility to pick and choose what is right for your processor design, but that flexibility creates a large number of possible combinations
- RISC-V Profiles specify a much smaller set of ISA choices that represent the most common use-cases
  - RVM for microcontrollers intended to run bare-metal code or an RTOS
  - RVA for application processors designed to run full operating system like Linux
- RISC-V Summit talk by Greg Favor [slides]

Learn more about RISC-V

- Get up-to-speed quick with the RISC-V Reader



RISC-V International

- The organization that develops the RISC-V specifications: **riscv.org**
- Non-profit with 2,700+ members including companies and universities
- Become a member - *free of cost to individuals and non-profits*
- RISC-V wiki has many helpful resources
- RISC-V mailing lists for many topics *(public archives)*
- Technical Meetings Calendar

RISC-V Open Hours

- Bi-weekly virtual meetup for the community to interact in real-time
  - Primary focus on RISC-V support in open source software and RISC-V dev boards
  - Call for participation is open! No prepared talk or slides required!
- Schedule
  - Wednesday, October 12, 7:00 PM (US PDT) which is Thursday morning in Asia
  - Wednesday, October 26, 9:00 AM (US PDT) which is early evening in Europe

“Is RISC-V an Open Source processor?”

- RISC-V is a set of specifications under an open source license
- RISC-V implementations can be open source or proprietary
- Open specifications make open source implementations possible
- **An open ISA makes it possible to design an open source processor**

RISC-V open source cores

- Academia
  - Rocket and BOOM from Berkeley, PULP family of cores from ETH Zurich
- Industry
  - SweRV created by Western Digital and now developed by CHIPS Alliance
  - OpenHW Group creating proven verified IP like their Core-V designs
  - Google OpenTitan silicon root of trust project uses LowRISC Ibex core
  - Alibaba T-Head released the OpenE902, OpenE906, OpenC906 (used in Allwinner D1 SoC) and OpenC910 cores on GitHub under Apache 2.0 license

RISC-V software ecosystem

- RISC-V already has a well supported software ecosystem
  - RISC-V International software committee coordinates efforts of member organizations
  - RISC-V extension and feature support
  - PLCT Lab led by Wei Wu at ISCAS does a lot of compiler and runtime work
- Operating systems: Linux, BSDs, FreeRTOS, Zephyr
- Toolchains & libraries: gcc, glibc, gdb, binutils, clang/llvm, newlib
- Languages and Runtimes: V8, Node.js, Rust, Go, OpenJDK, Python

RISC-V Privileged Architecture

- Three privilege modes
  - User (U-mode): application
  - Supervisor (S-mode): OS kernel
  - Machine (M-mode): firmware
- Environment Call (`ECALL`) instruction
  - Transfer control to a higher privileged mode
  - Userspace program (u-mode) uses ECALL to make system call into OS kernel (s-mode)

Control and Status Registers (CSRs)

- CSR have their own dedicated instructions to read and write
- CSR are specific to a mode (e.g. m-mode and s-mode)
- Machine Status (`mstatus`) is an important CSR

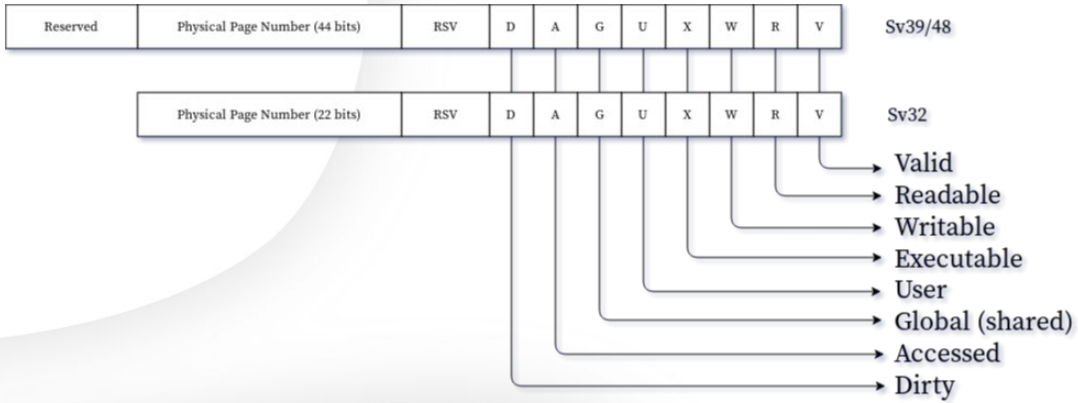
Bits	Field Name	Description
0	UIE	User Interrupt Enable
1	SIE	Supervisor Interrupt Enable
2	Reserved	
3	MIE	Machine Interrupt Enable
4	UPIE	User Previous Interrupt Enable
5	SPIE	Supervisor Previous Interrupt Enable
6	Reserved	
7	MPIE	Machine Previous Interrupt Enabler
8	SPP	Supervisor Previous Privilege
[10:9]	Reserved	
[12:11]	MPP	Machine Previous Privilege

Bits	Field Name	Description
[14:13]	FS	Floating Point State
[16:15]	XS	User Mode Extension State
17	MPRIV	Modify Privilege (access memory as MPP)
18	SUM	Permit Supervisor User Memory Access
19	MXR	Make Executable Readable
20	TVM	Trap Virtual memory
21	TW	Timeout Wait (traps S-Mode wfi)
22	TSR	Trap SRET
[23:30]	Reserved	
[31]	SD	State Dirty (FS and XS summary bit)

(source: Introduction to the RISC-V Architecture [PDF], Drew Barbier)

RISC-V Virtual Memory

- `satp` CSR (Supervisor Address Translation and Protection) controls supervisor-mode address translation and protection
- Sv32: 3 level page table
- Sv39: 3 level page table
- Sv48: 4 level page table
- Sv57: 5 level page table



(source: [Demystifying the RISC-V Linux software stack](), Nick Kossifidis)

RISC-V Trap Handling

- Exceptions occur synchronously
- Interrupts occur asynchronously
- `<x>cause` CSR indicates which interrupt or exception occurred
  - `mcause` for m-mode / `scause` for s-mode
- Corresponding bit is set in `<x>E/IP` CSR

Trap code[62:0]	Exception (Cause[MSB]=0)	Interrupt (Cause[MSB]==1)
0	Instruction addr misaligned	User Software Interrupt
1	Instruction access fault	Supervisor Software Interrupt
2	Illegal instruction	Reserved
3	Breakpoint	Machine Software Interrupt
4	Load address misaligned	User Timer Interrupt
5	Load access fault	Supervisor Timer Interrupt
6	Store/AMO addr misaligned	Reserved
7	Store/AMO access fault	Machine Timer Interrupt
8	Environment call	User External Interrupt
9	Reserved	Supervisor External Interrupt
10		Reserved
11		Machine External Interrupt
12	Instruction page fault	Reserved
13	Load page fault	
14	Reserved	
15	Store/AMO page fault	
>=16	Reserved	Reserved

(source: RISC-V Privileged Architecture [PDF], Allen Baum)

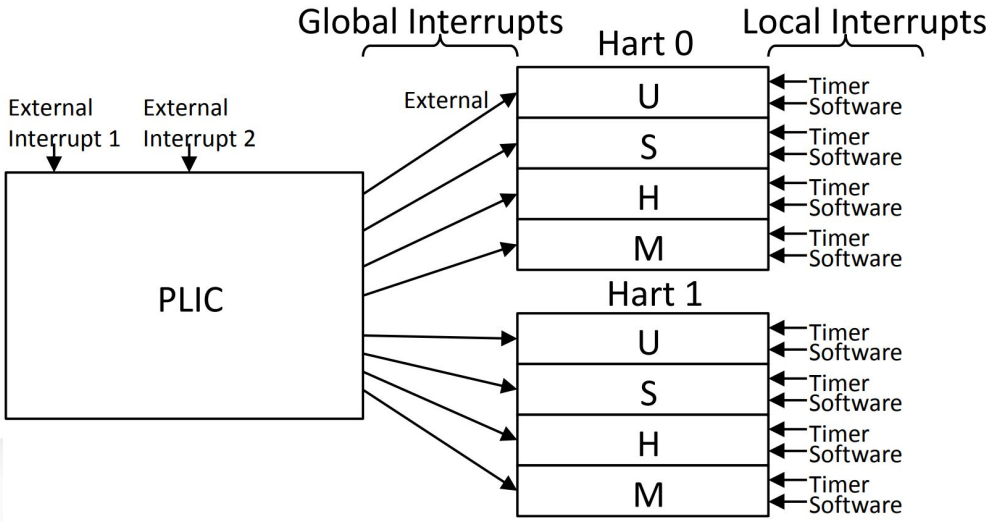
What is a Hart?

- Hart is a **har**dware **t**hread
- Each RISC-V core contains an independent instruction fetch unit
- A RISC-V core with multi-threading (SMT) would contain multiple harts
- Each hart is a processor from the perspective of Linux
  - Imagine a RISC-V laptop which has 2 cores with 2 harts per core
  - Linux would see 4 processors

(source: Section 1.1 in RISC-V Unprivileged spec)

RISC-V Interrupts

- Local per-hart interrupts
  - CLINT (Core Local Interruptor)
- Global interrupts
  - PLIC (Platform Level Interrupt Controller)

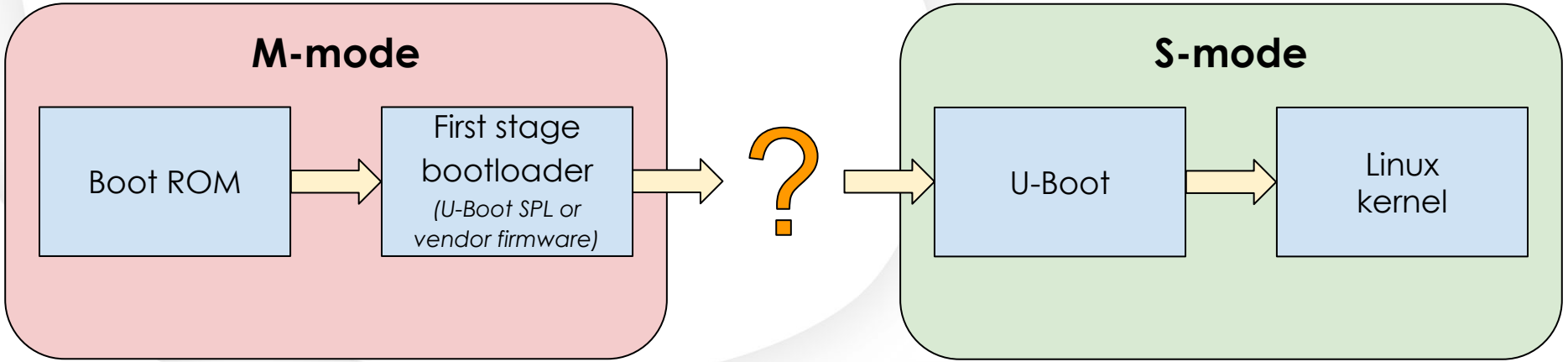


(source: RISC-V Fast Interrupts [PDF], Krste Asanovic)

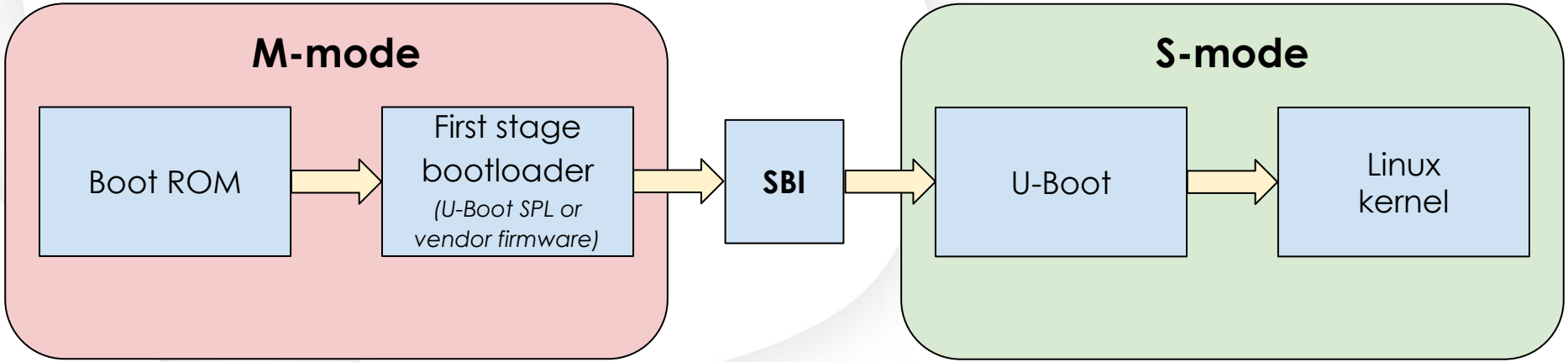
Advanced Interrupt Architecture (AIA)

- Developed on the AIA SIG mailing list: tech-aia
- **APLIC** (Advanced Platform-Level Interrupt Controller) replaces PLIC
- Adds **IMSIC** (Incoming Message-Signaled Interrupt Controller) for PCIe
- AIA is complimented by **ACLINT** (Advanced Core Local Interruptor)
  - Backwards compatible with the CLINT but restructured to be more efficient
  - RISC-V Summit talk by Anup Patel and John Hauser [slides]

RISC-V Boot Flow

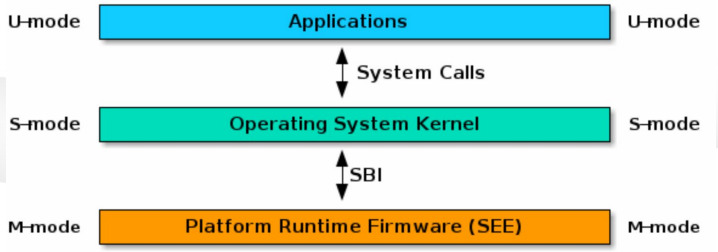


RISC-V Boot Flow



Supervisor Binary Interface (SBI)

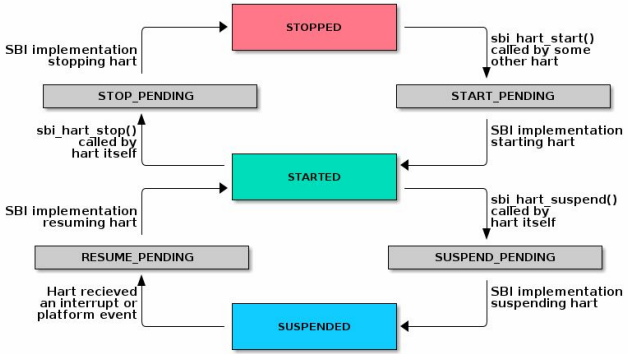
- Non-ISA RISC-V specification
  - This means it does not add or modify and RISC-V instructions
- The calling convention between S-mode and M-mode
  - Allows supervisor-mode (s-mode) software like the Linux to be portable across RISC-V implementations by abstracting platform specific functionality



Supervisor Binary Interface (SBI)

- Originally required by the UNIX-Class Platform Specification
  - Mailing list: tech-unixplatformspec
  - Now controlled by the Platform Runtime Services (PRS) Task Group
- Small core along with a set of optional modular extensions
  - Base extension - query basic information about the machine
  - Timer extension - program the clock for the next event
  - IPI extension - send an inter-processor interrupt to harts defined in mask
  - RFENCE extension - instructs remote harts to execute FENCE.I instruction

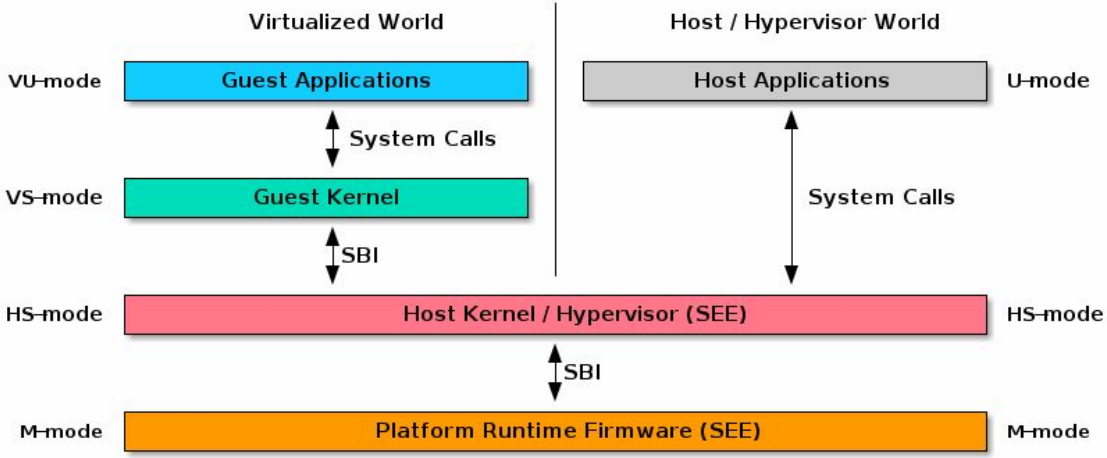
SBI Extensions



- Hart State Management (HSM)
  - S-mode can request to stop, start or suspend a hart
- System Reset
  - Supervisor-mode software can request system-level reboot or shutdown
- Performance Monitoring Unit
  - Interface for supervisor-mode to configure and use the RISC-V hardware performance counters with assistance from the machine-mode
  - "Performance Monitoring in RISC-V using perf" by Atish Patra

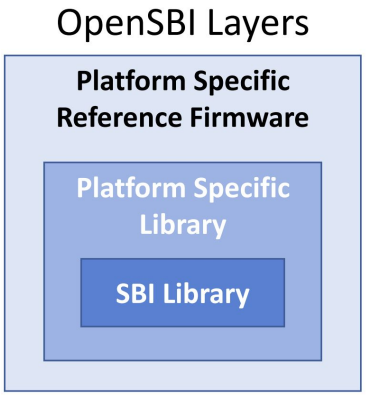
Hypervisor extension

- Hypervisor Supervisor mode (HS-mode) where host kernel runs
- Virtualized Supervisor mode (VS-mode) where the guest kernel runs



OpenSBI

- Open source implementation of SBI
  - Core library
  - Platform specific libraries
  - Full reference firmware for some platforms
- Provides runtime services to S-mode software
  - SBI extensions present on a platform define the available runtime services
  - Unimplemented instructions will trap and OpenSBI can emulate



(source: OpenSBI Deep Dive, Anup Patel)

OpenSBI Generic Platform

- No need to add code to OpenSBI for each new platform
  - First-stage bootloader, like U-Boot SPL, is expected to pass a Device Tree to OpenSBI which describes all the platform specific functionality
- The same OpenSBI binary can be used across platforms
  - Many RISC-V boards and emulators now use Generic Platform
  - Linux distros do not need to ship a different OpenSBI build for each board

UEFI Support

- UEFI is a standard interface between firmware and operating systems, and it is used on most x86 and arm64 platforms
- U-Boot and TianoCore EDK2 both have UEFI implementations on RISC-V
- Grub2 can be used as an UEFI payload on RISC-V
- UEFI support for RISC-V added in Linux 5.10

UEFI Support

- Boot hart ID is known only at boot and it is needed before ACPI tables or DT properties can be parsed
- Hart ID is passed in the a0 register on non-UEFI systems, but the UEFI application calling conventions do not allow this
- RISC-V EFI Boot Protocol allows the OS to discover the boot hart ID
- The public review process has completed, and Sunil V L has added support to the Linux kernel for RISCV_EFI_BOOT_PROTOCOL

RISC-V Platform Specification

- Goal is to support “off-the-shelf” software by standardizing the interface between hardware platforms and operating systems
- OS-A Platform
  - “A” as in application, this is a category of platforms that support full OS like Linux
  - OS-A Common Requirements
  - OS-A Embedded Platform
  - OS-A Server Platform

RISC-V Platform Specification

- OS-A common requirements for Embedded and Server
  - Must comply with the RVA22U and RVA22S ISA profiles as defined in RISC-V ISA Profiles
  - Common requirements for Debug, Timers, Interrupt Controllers
  - Requires serial console with UART 8250 or UART 16550
  - Requirements for runtime services such as SBI extensions
  - Software components must comply with the RISC-V Calling Convention specification and the RISC-V ELF specification

RISC-V Platform Specification

- OS-A Embedded Platform
  - Target might be a single board computer or mobile device
  - PMU counters and events for performance monitoring
  - Boot process must comply with Embedded Base Boot Requirements (EBBR) spec
  - EBBR requires a subset of the UEFI spec which U-Boot has implemented
  - Device Tree (DT) is the required mechanism for the hardware discovery and config
  - GPT partitioning required for shared storage

RISC-V Platform Specification

- OS-A Server Platform
  - Goal is for an enterprise Linux distro like RHEL to “just work” on server-class hardware that complies with this
  - System peripheral requirements like PCIe, watchdog timer, system date/time
  - RAS (Reliability, Availability, and Serviceability) requirements like ECC RAM
  - ACPI is the required mechanism for the hardware discovery and configuration
  - Must comply with the RISC-V ACPI Platform Requirements Specification

RISC-V ACPI Platform Specification

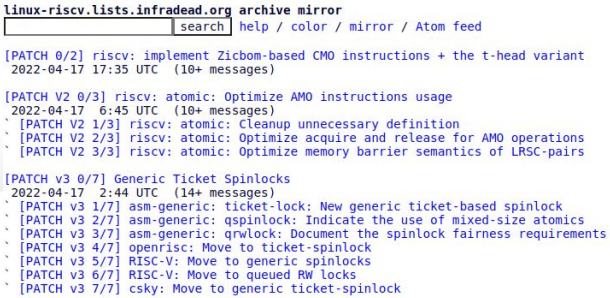
- Defines mandatory ACPI tables and objects for RISC-V server platforms
- New tables are needed for RISC-V
  - RISC-V Hart Capabilities Table (RHCT)
  - RISC-V Timer Description Table (RTDT)
- More details in '[ACPI for RISC-V: Enabling Server Class Platforms]'
  - Sunil V L from Ventana Microsystems at RISC-V Summit [slides]

RISC-V emulation in QEMU

- Support for RISC-V in mainline QEMU
- Boots 32-bit and 64-bit mainline Linux kernel
- Machine configs to boot same binaries as some RISC-V dev boards
- Supports the new Hypervisor and Vector extensions

RISC-V in the Linux kernel

- Initial port by Palmer Dabbelt was merged into Linux 4.15 back in 2018
- Palmer continues to maintain the riscv tree
- Development happens on the linux-riscv mailing list
- View the archive on lore.kernel.org
- IRC: #riscv on libera.chat



Added in the past year

- KVM RISC-V support by Anup Patel added in Linux 5.16
  - Add KVM support for the Hypervisor specification
- SBI SRST extension support by Anup Patel added in Linux 5.17
  - Support for the SBI SRST (system reset) extension which allows systems that do not have an explicit driver in Linux to reboot

Linux 5.18

- Add Sv57 page table support by Qinglin Pan
    - use 5-level page table to support Sv57 which expands the virtual address space to 57 bits (128 petabytes)
- Improve RISC-V Perf support by Atish Patra
    - Recent talk: Perf on RISC-V: The Past, the Present and the Future (slides)

Linux 5.18

- RISC-V CPU Idle support by Anup Patel
  - cpuidle and suspend drivers now support the SBI HSM extension
- Provide framework for RISC-V ISA extensions by Atish Patra
  - Linux was no longer correctly parsing the RISC-V ISA string as the number of RISC-V extensions has grown and extension names are no longer a single character
  - This series implements a generic framework to parse multi-letter ISA extensions.
  - Based on initial work by Tsukasa OI

Linux 5.19

- Support for page-based memory attributes
  - *We'll dive into that topic in a few slides...*
- Support for running rv32 binaries on rv64 systems via the compat subsystem. This is useful for systems with very limited RAM.
- Support new generic ticket-based spinlocks, which allows us to also move to qrwlock
- Support for kexec_file()

Upcoming Linux 6.0

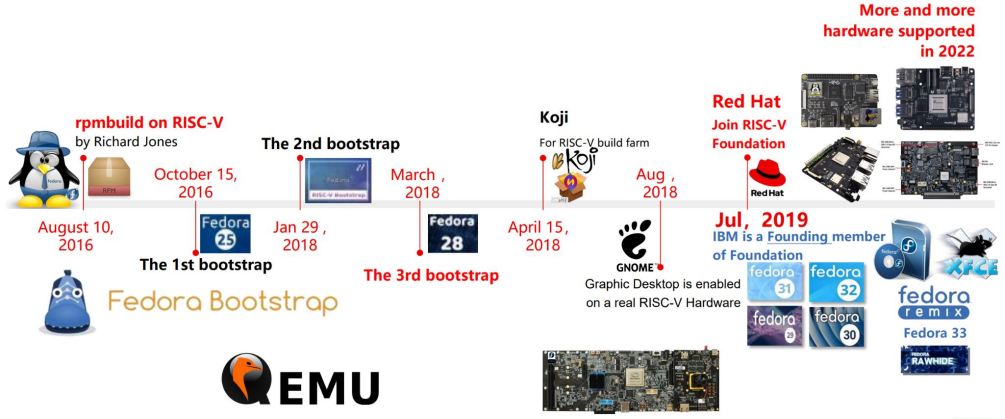
- Pull requests originally sent for 5.20 but Linus decided to call it 6.0
- [PATCH v7 0/4] Add Sstc extension support by Atish Patra
  - Traditionally, an SBI call is necessary to generate timer interrupts as S-mode does not have access to the M-mode time compare registers.
  - This results in significant latency for the kernel to generate timer interrupts at kernel
  - For virtualized environments, it's even worse as the KVM handles the SBI call and uses a software timer to emulate the time compare register.
  - Sstc extension allows kernel to program a timer and receive interrupt without supervisor execution environment (M-mode/HS mode) intervention.

Work in progress

- [PATCH v10 00/16] riscv: Add vector ISA support by Greentime Hu
  - Vector ISA support based on the ratified Vector 1.0 extension
  - Defines new structure `__riscv_v_state` in struct `thread_struct` to save/restore the vector related registers. It is used for both kernel space and user space.
- [PATCH v9 0/7] RISC-V IPI Improvements by Anup Patel
  - Traditionally, RISC-V S-mode software like the Linux kernel calls to into M-mode runtime firmware like OpenSBI to issue IPIs (inter-processor interrupts)
  - AIA (advanced interrupt architecture) provides the ability for S-mode to issue IPIs without any assistance from M-mode. This improves efficiency.

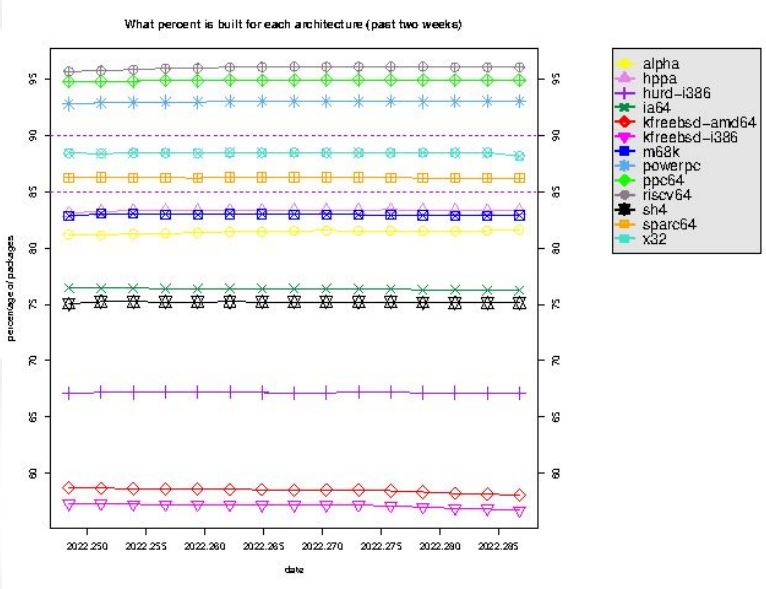
Linux distro: Fedora

- Aims to provide a complete Fedora experience on RISC-V
- Talk by Wei Fu, RISC-V Ambassador and Red Hat engineer [slides]
- Installation instructions for QEMU and RISC-V dev boards



Linux distro: Debian

- riscv64 is port of Debian to RISC-V
  - “a port in Debian terminology means to provide the software normally available in the Debian archive (over 20,000 source packages) ready to install and run”
- 95% of packages are built for RISC-V
  - The Debian port uses RV64GC as the hardware baseline and the lp64d ABI



Linux distro: Ubuntu

- riscv64 supported since the release of Ubuntu 20.04 LTS.
- Ubuntu 22.04 pre-installed SD-card image for several boards & QEMU
- Starting with Ubuntu 22.04, a server install image is made available to install Ubuntu on NVMe drive of the SiFive Unmatched board

Linux distros

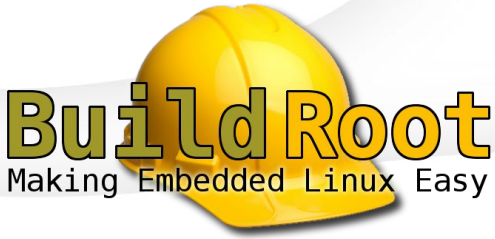
- OpenSuSE
  - RISC-V support is still under development with Tumbleweed images for some boards
- Arch Linux
  - Community effort has 95% of core packages building for RISC-V
- Gentoo
  - riscv64 stages are available on the Gentoo download page

OpenEmbedded and Yocto

- meta-riscv: general hardware-specific BSP overlay for RISC-V devices
  - works with different OpenEmbedded/Yocto distributions and layer stacks
  - Supports both QEMU and RISC-V dev boards

BuildRoot

- RISC-V port is now supported in the upstream BuildRoot project
- "Embedded Linux from scratch in 45 minutes (on RISC-V)"
  - Tutorial by Michael Opdenacker at FOSDEM 2021
  - Use Buildroot to compile OpenSBI, U-Boot, Linux and BusyBox
  - Boot the system in QEMU



Allwinner D1 SoC

- Mass production low cost SoC with a single T-Head C906 core at 1 GHz
- Allwinner reused peripheral IP from their existing ARM SoCs and the Linux kernel already has drivers for most of them
- However, T-Head MMU has a non-standard ‘enhanced’ mode that is needed to support DMA with devices on non-coherent interconnects
- Linux needs to enable that ‘enhanced’ MMU mode to function

T-Head PTE format

- T-Head used bits in the PTE (Page Table Entry) to specify memory type

```
| 63 | 62 | 61 | 60 | 59-8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0
  SO   C    B    SH   RSW    D   A   G   U   X   W   R   V
  ^    ^    ^    ^

BIT(63): SO - Strong Order
BIT(62): C  - Cacheable
BIT(61): B  - Bufferable
BIT(60): SH - Shareable

0000 - NC   Weakly-ordered, Non-cacheable, Non-bufferable, Non-shareable
0111 - PMA  Weakly-ordered, Cacheable, Bufferable, Shareable
1000 - IO   Strongly-ordered, Non-cacheable, Non-bufferable, Non-shareable
```

… but those bits were already marked reserved in RISC-V priv spec

Page-Based Memory Types extension

- **Svpbmt** proposed by Virtual Memory TG and ratified at the end of 2021
  - *"S" = supervisor-mode (privileged architecture), "v" for virtual memory*

```
Here is the svpbmt PTE format:
| 63 | 62-61 | 60-8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0
  N     MT     RSW    D   A   G   U   X   W   R   V
        ^

RISC-V
Encoding &
MemType     RISC-V Description
----------  ------------------------------------------------
00 - PMA    Normal Cacheable, No change to implied PMA memory type
01 - NC     Non-cacheable, idempotent, weakly-ordered Main Memory
10 - IO     Non-cacheable, non-idempotent, strongly-ordered I/O memory
11 - Rsvd   Reserved for future standard use
```

Svpbmt support in Linux

- [PATCH v10 00/12] riscv: support for Svpbmt and D1 memory types
  - by Heiko Stuebner based on initial work by Alibaba kernel engineer Guo Ren
  - Implements the official RISC-V Svpbmt extension
  - The standard Svpbmt and custom T-Head PTE formats both use the highest bits to determine memory type but the encoding and semantics are different
  - T-Head PTE format is supported through boot-time instruction patching using the Linux Alternatives Framework. Heiko presented on this topic at Embedded World 2022.
  - Patch series was merged and landed in Linux 5.19 release

Cache Management Operations

- Instructions to manage cache are important for SoCs which that lack cache coherent interconnects
- Zicbom extension *("Z" prefix means Unpriv spec)* was ratified at the end of 2021, and it defines cache-block management (CBO) instructions:
  - CBO.CLEAN guarantee store by hart can be read from mem by non-coherent device
  - CBO.INVAL guarantee hart can load data written to memory by non-coherent device
  - CBO.FLUSH guarantees both

Support for Non-Coherent I/O Devices in RISC-V from RV Summit [slides]

CMO support in Linux

- riscv: implement Zicbom-based CMO instructions + the t-head variant
  by Heiko Stuebner
- Implements Zicbom-extension to handle cache clean, invalidate, flush

```
* cbo.clean rs1
* | 31 - 20 | 19 - 15 | 14 - 12 | 11 - 7 | 6 - 0 |
*    0...01     rs1       010      00000  0001111
*
* cbo.flush rs1
* | 31 - 20 | 19 - 15 | 14 - 12 | 11 - 7 | 6 - 0 |
*    0...10     rs1       010      00000  0001111
*
* cbo.inval rs1
* | 31 - 20 | 19 - 15 | 14 - 12 | 11 - 7 | 6 - 0 |
*    0...00     rs1       010      00000  0001111

#define CBO_INVAL_A0   ".long 0x15200F"
#define CBO_CLEAN_A0   ".long 0x25200F"
#define CBO_FLUSH_A0   ".long 0x05200F"
```

CMO support in Linux

- T-Head implemented custom cache instructions before Zicbom existed

```
* dcache.ipa rs1 (invalidate, physical address)
* | 31 - 25 | 24 - 20 | 19 - 15 | 14 - 12 | 11 - 7 | 6 - 0 |
*   0000001    01010      rs1       000      00000  0001011
* dache.iva rs1 (invalida, virtual address)
*   0000001    00110      rs1       000      00000  0001011
*
* dcache.cpa rs1 (clean, physical address)
* | 31 - 25 | 24 - 20 | 19 - 15 | 14 - 12 | 11 - 7 | 6 - 0 |
*   0000001    01001      rs1       000      00000  0001011
* dcache.cva rs1 (clean, virtual address)
*   0000001    00100      rs1       000      00000  0001011
*
* dcache.cipa rs1 (clean then invalidate, physical address)
* | 31 - 25 | 24 - 20 | 19 - 15 | 14 - 12 | 11 - 7 | 6 - 0 |
*   0000001    01011      rs1       000      00000  0001011
* dcache.civa rs1 (... virtual address)
*   0000001    00111      rs1       000      00000  0001011

#define THEAD_INVAL_A0  ".long 0x0265000b"
#define THEAD_CLEAN_A0  ".long 0x0245000b"
#define THEAD_FLUSH_A0  ".long 0x0275000b"
```

CMO support in Linux

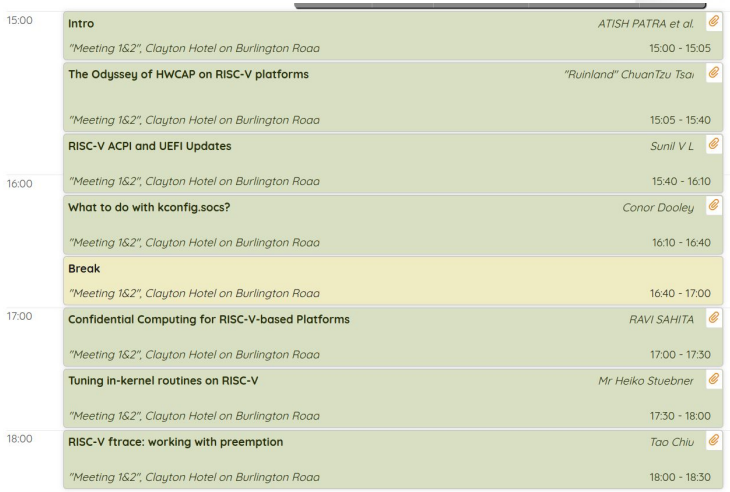
- While the Zicbom and T-Head instructions are different, they provide the same functionality, so the T-Head variant handled with the existing alternatives mechanism
- Allwinner D1 needs these cache instructions for peripherals like MMC (SD card), USB, and Ethernet to work
- Support will be in the Linux 6.0 release as Heiko's patch series was included in Palmer's 5.20* PR to Linus

arch/riscv patch acceptance guidelines

- The RISC-V instruction set architecture is developed in the open: in-progress drafts are available for all to review and to experiment with implementations. New module or extension drafts can change during the development process - sometimes in ways that are incompatible with previous drafts.
- This flexibility can present a challenge for RISC-V Linux maintenance.
- General rule: only frozen or ratified extensions will be supported in Linux
- This policy will be updated to be more specified after Linux Plumbers

RISC-V micro-conference

- During Linux Plumbers Conference earlier this week
- Live stream *(individual videos of talks posted later)*



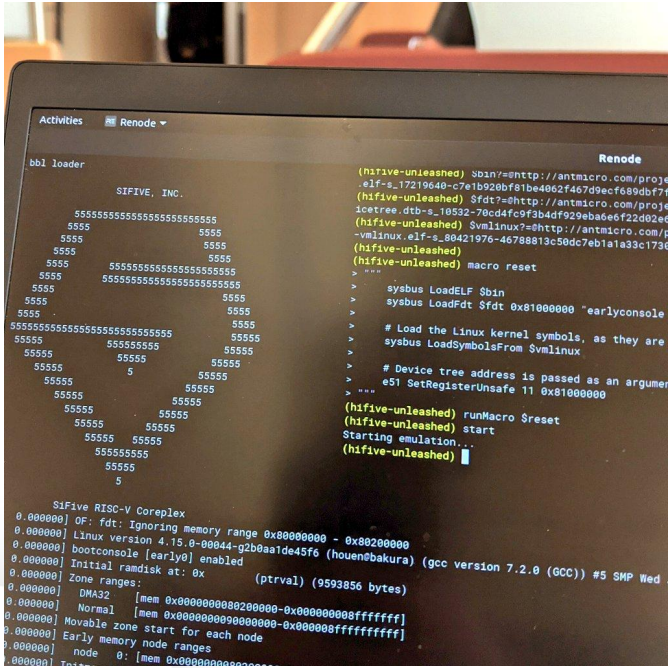
15:00	**Intro**	*ATISH PATRA et al.*	
	"Meeting 1&2", Clayton Hotel on Burlington Roaa		15:00 - 15:05
	The Odyssey of HWCAP on RISC-V platforms	*"Ruinland" ChuanTzu Tsai*	
	"Meeting 1&2", Clayton Hotel on Burlington Roaa		15:05 - 15:40
	RISC-V ACPI and UEFI Updates	*Sunil V L*	
16:00	*"Meeting 1&2", Clayton Hotel on Burlington Roaa*		15:40 - 16:10
	What to do with kconfig.socs?	*Conor Dooley*	
	"Meeting 1&2", Clayton Hotel on Burlington Roaa		16:10 - 16:40
	Break		
	"Meeting 1&2", Clayton Hotel on Burlington Roaa		16:40 - 17:00
17:00	**Confidential Computing for RISC-V-based Platforms**	*RAVI SAHITA*	
	"Meeting 1&2", Clayton Hotel on Burlington Roaa		17:00 - 17:30
	Tuning in-kernel routines on RISC-V	*Mr Heiko Stuebner*	
	"Meeting 1&2", Clayton Hotel on Burlington Roaa		17:30 - 18:00
18:00	**RISC-V ftrace: working with preemption**	*Tao Chiu*	
	"Meeting 1&2", Clayton Hotel on Burlington Roaa		18:00 - 18:30

RISC-V Developer Boards

- Initiative from RISC-V International to get Linux-capable boards into the hands of open source developers
  - Launched in 2021 with the Allwinner D1 Nezha and SiFive Unmatched
- Fill out this form to apply
  - Need to be RISC-V International member (or part of a member organization), but remember that individuals can join RISC-V International free of cost
  - Explain why you are interested in RISC-V and what you plan to do with dev board
  - To improve your chances, don't overestimate your hardware requirements like RAM

No hardware? Try Renode!

- Emulate physical hardware systems including CPU, peripherals, sensors, and networking
- Run the same binaries as the real hardware for over 30 supported dev boards
  - Microchip PolarFire SoC Icicle Kit
  - Kendryte K210
  - SiFive HiFive Unleashed



Slides: **tinyurl.com/elce22-riscv**

Let's talk more at the

RISC-V and Open Hardware BoF

at **3:55 PM** this afternoon

Liffey Meeting Room 2 (Level 1)

Drew Fustini <dfustini@baylibre.com>